An indicator for the switch from derivative-free to derivative-based optimization

S. Gratton [a], N. Soualmi [b], L.N. Vicente [c,*]

[a] ENSEEIHT, INPT, rue Charles Camichel, B.P. 7122 31071, Toulouse Cedex 7, France
[b] CERFACS, 42 Avenue Gaspard Coriolis, 31057 Toulouse Cedex 01, France
[c] CMUC, Department of Mathematics, University of Coimbra, 3001-501 Coimbra, Portugal

Keywords:

Derivative-free optimization
Derivative-based optimization
Indicators
Direct-search methods
Gradient methods Complexity and global rates

A B S T R A C T

In some optimization problems found in applications, the derivatives of the objective function can be computed or approximated but at an expensive cost, and it is desirable to know when to use derivative-free methods (such as direct search, for instance) or derivative-based methods (such as gradient or quasi-Newton methods). Derivative-free methods may achieve a steady initial progress for some problems, but after some advance they may also become slower or even stagnate due to the lack of derivatives. It is thus of interest to provide a way to appropriately switch from a derivative-free method to a derivative-based one. In this paper, we develop a family of indicators for such a switch based on the decrease properties of both classes of methods (typically used when deriving worst case complexity bounds).

1. Introduction and basic concepts

The calculation of functions involved in optimization problems appearing in computational sciences and engineering is frequently based on numerical simulation. The smoothness of the function and the access to whatever form of derivatives vary considerably across applications. While there are problems of totally black-box type where only function values can be computed in a certain (sometimes unknown) feasible region, there are other more structured problems of continuously differentiable type where both function values and gradients can be computed, an example of special interest to us being the acoustic full-waveform inverse problem in Earth imaging [11]. In such problems, the calculation of the gradient may come at a cost higher than the one for the function value, and such difference may depend on the dimension of the problem and the accuracy required.

Although Derivative-Free Optimization provides now a theory [1] to understand models and families of directions used in the various classes of methods as well as their convergence properties, a question that to our knowledge has never been addressed is when should one switch from a derivative-free method to a derivative-based one, when the gradient can be computed (or possibly approximated by finite differences). Such a general question can be posed in many different ways depending on the methods under consideration, their costs per iteration, the computational budget available, and the final accuracy desired for a solution. Other issues like non-smoothness or local vs global optimization may also play a relevant role when analyzing such an issue.

In this paper, we try to make a first contribution to the topic by considering a continuously differentiable setting where the gradient of the objective function $f : \mathbb{R}^n \to \mathbb{R}$ can be computed (or possibly approximated by finite differences), and the minimization of the function is unconstrained. The gradient of f will be considered Lipschitz continuous in $\mathbb{R}^n$ (or in a level set corresponding to an initial iterate), with constant $L_{\nabla f} > 0$. Our idea to develop an indicator for the switch under consideration will be to form appropriate ratios of lower bounds for the decreases attained in successful iterations (involving the derivative-free method and what would be expected for the derivative-based one). These lower bounds will be the ones used when deriving worst case complexity bounds (WCC) or global rates for such methods, and we will try to take advantage of the mismatches that appear in such bounds with or without using derivatives. Our ultimate goal is to detect a switch when not enough progress is being achieved compared to the one that could be done if derivatives were available, letting the derivative-free method continue otherwise (meaning to do not switch).

To illustrate and test our ideas, we will consider the following simple direct-search method which imposes a sufficient decrease condition based on a quadratic forcing function. We consider an

* Corresponding author.
E-mail addresses: serge.gratton@enseeiht.fr (S. Gratton), nacer.soualmi@cerfacs.fr (N. Soualmi), lnv@mat.uc.pt (L.N. Vicente).

http://dx.doi.org/10.1016/j.orl.2017.05.003

iteration uniquely defined by a poll step which evaluates the objective function using a positive spanning set (PSS), i.e., a set of non-zero vectors that spans $\mathbb{R}^n$ with non-negative coefficients.

Algorithm 1.1. Direct-search method (polling)

Initialization: Choose a PSS D, an initial point x_0, and an initial step size $\alpha_0 > 0$. The constants $0 < \beta < 1 \leq \gamma$ are specified. Set $k = 0$.

1. Poll step: Order the set of poll points $P_k = \{x_k + \alpha_k d : d \in D\}$. Start evaluating f at the poll points following the chosen order. If a poll point $x_k + \alpha_k d$ is found such that $f(x_k + \alpha_k d_k) < f(x_k) - \alpha_k^2/2$, then set $x_{k+1} = x_k + \alpha_k d_k$ and declare the iteration successful. Otherwise, declare the iteration unsuccessful and set $x_{k+1} = x_k$.

2. Update iterate and step size: If the iteration was successful, then maintain or increase the step size parameter: $\alpha_{k+1} \in [\alpha_k, \gamma\alpha_k]$. Otherwise, decrease the step size parameter $\alpha_{k+1} = \beta\alpha_k$. Increment k by one and go to Step 1.

It is well known that such an algorithm is well defined (for functions with Lipschitz continuous gradients) in the sense that a successful iteration is found in a finite number of step-size reductions [7]. In fact, it is possible to prove [7] that if the iteration k is unsuccessful, then

$$\|\nabla f(x_k)\| \leq \mathrm{cm}(D)^{-1}\left(L_{\nabla f}\frac{\max_{d\in D}\|d\|}{2} + \frac{1}{2\min_{d\in D}\|d\|}\right)\alpha_k, \quad (1)$$

where $\mathrm{cm}(D)$ is the cosine measure of the PSS D, defined as

$$\mathrm{cm}(D) = \min_{0\neq v\in\mathbb{R}^n}\max_{d\in D}\frac{v^\top d}{\|v\|\|d\|}.$$

The cosine measure of a PSS is always positive. For instance, the PSS $D_\oplus$ formed by the coordinate vectors and their negatives is such that $\mathrm{cm}(D_\oplus) = 1/\sqrt{n}$, and so is $QD_\oplus$ where Q is an orthogonal matrix. Given an $\epsilon \in (0, 1)$, it is known that such an algorithm takes at most $\mathcal{O}(n\epsilon^{-2})$ iterations and $\mathcal{O}(n^2\epsilon^{-2})$ function evaluations to drive the norm of the gradient below ϵ (see [10]).[1] The dependence of these WCC bounds on ϵ reduces to ϵ^{-1} if the function is convex and to $-\log(\epsilon)$ if the function is strongly convex (see [2]). These bounds depend quadratically on the Lipschitz constant $L_{\nabla f}$.

For the sake of simplicity, we take the gradient method with backtracking as our derivative-based method.

Algorithm 1.2. Gradient method (backtracking)

Initialization: Choose initial point x_0. Let $c \in (0, 1)$ and $b > 0$ be specified. Set $k = 0$.

1. Backtrack: Let α_k be the first scalar in $b, b/2, b/4, \ldots$ such that

$$f(x_k - \alpha_k\nabla f(x_k)) \leq f(x_k) - c\alpha_k\|\nabla f(x_k)\|^2. \quad (2)$$

2. Update iterate: Compute $x_{k+1} = x_k - \alpha_k\nabla f(x_k)$. Increment k by one and go to Step 1.

It is known that Algorithm 1.2 is well defined in the sense that it is always possible to find α_k of the form given in the algorithm such that (2) is satisfied (see, e.g., [9]). Moreover, each iteration of Algorithm 1.2 satisfies

$$f(x_k) - f(x_{k+1}) \geq C\|\nabla f(x_k)\|^2, \quad (3)$$

with

$$C = c\max\left(\frac{1-c}{L_{\nabla f}}, b\right). \quad (4)$$

[1] The notation $\mathcal{O}(A)$ will mean a scalar times A, where the scalar does not depend on the iteration counter of the method under analysis (thus depending only on the problem or on algorithmic constants).

It is also well known (see [8]) that the WCC effort for the gradient method (to reduce the norm of the gradient below ϵ) is of $\mathcal{O}(\epsilon^{-2})$ in general, reducing to $\mathcal{O}(\epsilon^{-1})$ and $\mathcal{O}(-\log(\epsilon))$ in the convex and strongly convex cases, respectively. Note that these bounds depend linearly on $L_{\nabla f}$.

The remaining of this paper is organized in three sections. In Section 2, we will describe our main idea to develop a family of indicators for the switch from derivative-free to derivative-based iterations using direct search and the gradient method as motivation. Two concrete indicators are then proposed in Section 3 and their numerical performance (using Algorithms 1.1 and 1.2) is reported. Finally, in Section 4, we will further discuss the scope of our approach.

2. Elements for the indicators

At each iteration of a gradient-based method, one typically has

$$f(x_k) - f(x_{k+1}) \geq \frac{G}{L_{\nabla f}}\|\nabla f(x_k)\|^2, \quad (5)$$

where $G > 0$ is a fixed constant independent of f or of the iteration counter. This is the case for the gradient method with line search satisfying both Wolfe conditions (sufficient decrease condition (2) and curvature condition), or for the gradient method with backtracking line search imposing only the sufficient decrease condition (2); (see Algorithm 1.2 and (3)–(4)).

At each successful iteration of a derivative-free method based on sufficient decrease, one typically has

$$f(x_k) - f(x_{k+1}) \geq D_1\, t_k^2,$$

where $D_1 > 0$ is a fixed algorithmic parameter independent of f and of the iteration counter. The step-size parameter t_k represents the trust-region radius δ_k in derivative-free trust-region methods or the step size α_k in direct-search methods (see Algorithm 1.1). If ℓ is an iteration where the step-size parameter t_ℓ is reduced (for either class of methods), one has

$$\|\nabla f(x_\ell)\| \leq D_2(n)L_{\nabla f}\, t_\ell, \quad (6)$$

where $D_2(n) > 0$ is a fixed constant independent of f and of the iteration counter (but typically dependent on n); see (1) for the direct-search case and [3] for the trust-region one.

Let k be a given successful iteration and r_k the last iteration before k where the step size has been reduced. Let C_k be the set of indices corresponding to successful iterations between r_k and k where some approximation to the gradient is known.

The decrease produced by a gradient-based method would have been at least

$$f(r_k) - f(x_{k+1}) \geq \sum_{j\in C_k}\frac{G}{L_{\nabla f}}\|\nabla f(x_j)\|^2. \quad (7)$$

On the other hand, the decrease produced by a derivative-free method would have been at least

$$f(r_k) - f(x_{k+1}) \geq \sum_{j\in C_k} D_1 t_j^2 \geq \frac{D_1|C_k|}{D_2(n)^2 L_{\nabla f}^2}\|\nabla f(x_{r_k})\|^2. \quad (8)$$

Establishing a ratio between the decreases in (8) with those in (7), yields two quantities

$$\frac{D_1 L_{\nabla f}\sum_{j\in C_k} t_j^2}{G\sum_{j\in C_k}\|\nabla f(x_j)\|^2} \quad \text{and} \quad \frac{D_1|C_k|\|\nabla f(x_{r_k})\|^2}{GD_2(n)^2 L_{\nabla f}\sum_{j\in C_k}\|\nabla f(x_j)\|^2}.$$

This motivates the introduction of the following two indicators

$$I_k^1 = \frac{\sum_{j\in C_k} t_j^2}{\sum_{j\in C_k}\|g_j\|^2} \quad \text{and} \quad I_k^2 = \frac{|C_k|\|g_{r_k}\|^2}{\sum_{j\in C_k}\|g_j\|^2} \quad (9)$$

for the switch from the derivative-free regime to the derivative-based one, where g_i represents an approximation for $\nabla f(x_i)$. Such approximations can be computed based on previous evaluations of the objective function (or in a last resource using finite differences), as we explain later in the paper.

These indicators should be in principle increasing in k. In the I^1 case, this is because the gradient is expected to go to zero faster than the step size (see (6)). In the I^2 case, this is because the gradient in the derivative case is expected to go to zero faster than in the derivative-free one. A possible switch is therefore when such indicators become larger than a specified positive threshold.

Note that we omitted the constant $D_2(n)^2$ in (9) but such a constant could have been kept in the indicator I_k^2 since it scales with n and thus provides some relevant information. Also, we have omitted the unknown Lipschitz constant, although it should be noticed that this constant appears differently in both indicators.

3. Two concrete indicators and their numerical performance

A number of issues have to be addressed to make such indicators of practical use.

3.1. Two concrete indicators

First, a switch based on the size of an indicator has to be made relatively to its scaling. So, one has to compute an initial indicator of reference I_{scaling} (see Algorithm 3.2) that can be then used as a scaling factor to determine the moment of the switch. Such a switch will occur when $I_k/I_{\text{scaling}} \geq C_{\text{threshold}}$, for a certain positive threshold $C_{\text{threshold}}$ (see Algorithm 3.1).

Our first concrete indicator is solely based on I_k^1 in (9). The rationale here is that I_k^1 is derived from a tighter bound in (8). The computation of the indicator and of the scaling factor I_{scaling} for this pure I^1 case is given in Algorithms 3.1 and 3.2, respectively. In the definition of I_k^1 one takes into account all successful iterations from k until the last unsuccessful one r_k. Hence, one has to take provision for the fact that one can make a long series of successful iterations, and that such an event may occur from the initial iteration. Algorithm 3.3 (pure I^1) takes these aspects into consideration by using a maximum prespecified number E of successful iterations for backtracking from k.

We assume that we are using Direct Search (Algorithm 1.1) as our derivative-free method (DF-Method) and the Gradient Method (Algorithm 1.2) as our derivative-based one (DB-Method), but the presentation is sufficiently abstract to be independent of the specifics of these methods as long as they conform with the presentation of Section 2. After the algorithm descriptions of the indicators, we discuss the calculation of the gradient approximations mentioned there.

Algorithm 3.1. Procedure for computing the (pure) indicator I^1 at a successful iteration $k \geq E$. It is assumed that the kth iteration of the DF-Method has been performed and was successful. It requires the constants E, I_{scaling}, and $C_{\text{threshold}}$.

Set r_k = index of last previous unsuccess ($r_k = -\infty$ if it does not exist).
Set $C_k = \{\max(r_k + 1, k - E), \ldots, k\}$.
Set $I_k = \frac{\sum_{j\in C_k} t_j^2}{\sum_{j\in C_k} \|g_j\|^2}$.
if $\frac{I_k}{I_{\text{scaling}}} \geq C_{\text{threshold}}$ **then**
  Switch to the DB–Method.
else
  Continue the calculation of the indicator in the DF–Method.
end if

Now, we present a possible way to compute the scaling factor I_{scaling}.

Algorithm 3.2. Procedure for computing the scaling factor I_{scaling} for the (pure) indicator I_1. It is assumed that the first $E_{\text{scaling}} (\leq E)$ iterations of the DF-Method have been run.

Set r_k = the last unsuccessful iteration ($r_k = -1$ if all the first E_{scaling} are successful).
Set $C_k = \{r_k + 1, \ldots, E_{\text{scaling}} - 1\}$.
if $C_k \neq \emptyset$ **then**
  Set $I_{\text{scaling}} = \frac{\sum_{j\in C_k} t_j^2}{\sum_{j\in C_k} \|g_j\|^2}$.
else
  Run DF–Method until a successful iteration k_s is performed.
  Set $I_{\text{scaling}} = \frac{\|t_{k_s}^2\|}{\|g_{k_s}\|^2}$.
end if

In our second concrete indicator (Algorithm 3.3), we use the indicator I_k^2 in (9) as much as possible as it relates comparable information with or without using derivatives. However, such an indicator requires explicitly a previous unsuccessful iteration. Also here, one has to take provision for the fact that one can make a long series of successful iterations, and that such an event may occur from the initial iteration. Algorithm 3.3 (hybrid I^1/I^2) takes these aspects into consideration by using a prespecified number E of successful iterations after which we resort to using I_k^1 in (9) instead.

Algorithm 3.3. Procedure for computing the hybrid indicator I^1/I^2 at a successful iteration $k \geq E$. It is assumed that the kth iteration of the DF-Method has been performed and was successful. It requires the constants E, I_{scaling}, and $C_{\text{threshold}}$.

Set r_k = index of last previous unsuccess ($r_k = -\infty$ if it does not exist).
if ($k > E$ and $r_k = -\infty$) or ($k - r_k > E$) **then**
  Set $C_k = \{k - E, \ldots, k\}$.
  Set $I_k = \frac{\sum_{j\in C_k} t_j^2}{\sum_{j\in C_k} \|g_j\|^2}$.
else
  Set $C_k = \{r_k + 1, \ldots, k\}$.
  Set $I_k = \frac{|C_k|\|g_{r_k}\|^2}{\sum_{j\in C_k} \|g_j\|^2}$.
end if
if $\frac{I_k}{I_{\text{scaling}}} \geq C_{\text{threshold}}$ **then**
  Switch to the DB–Method.
else
  Continue the calculation of the indicator in the DF–Method.
end if

Algorithm 3.4. Procedure for computing the scaling factor I_{scaling} for the hybrid indicator I^1/I^2. It is assumed that the first $E_{\text{scaling}} (\leq E)$ iterations of the DF–Method have been run.

if the first E_{scaling} iterations are successful **then**
  Set $C_k = \{0, \ldots, E_{\text{scaling}} - 1\}$.
  Set $I_{\text{scaling}} = \frac{\sum_{j\in C_k} t_j^2}{\sum_{j\in C_k} \|g_j\|^2}$.
else
  if the E_{scaling}-th iteration is successful **then**
    Set r_k as the index of the last unsuccessful iteration.
    Set $I_{\text{scaling}} = \frac{|C_k|\|g_{r_k}\|^2}{\sum_{j\in C_k} \|g_j\|^2}$.
  else
    Run DF–Method until a successful iteration k_s is performed.
    Set $I_{\text{scaling}} = \frac{\|g_{k_s-1}\|^2}{\|g_{k_s}\|^2}$.
  end if
end if

These procedures to calculate indicators for the switch to derivative-based optimization may require the use of approximations to the gradient. There are different ways of estimating the gradient when using a derivative-free method. If one applies a trust-region method, one can use the gradients of the models.

When using a direct-search method as we do in our paper, one can compute simplex gradients based on the previous evaluated points. Given a sample set $Y = \{y^0, \dots, y^p\}$, a simplex gradient is nothing else than the gradient g of a linear interpolation or regression model $m(y; y^0) = f(y^0) + g^\top(y - y^0)$ centered at one of the points $y^0 \in Y$ (typically $y^0 = x_k$). Let the interpolating conditions $m(y^i; y^0) = f(y^i), i = 1, \dots, p$ be written as $L^\top g = \delta f(Y)$ where

$$L = \left[y^1 - y^0 \cdots y^n - y^0\right]^\top \quad \text{and} \quad \delta f(Y) = \begin{bmatrix} f(y^1) - f(y^0) \\ \vdots \\ f(y^n) - f(y^0) \end{bmatrix}.$$

Assume that the columns of L are linearly independent. Then, the simplex gradient is computed as $g = \nabla_s f(y^0) = L^{-1}\delta f(Y)$ if $p = n$ (determined case) or as $g = \nabla_s f(y^0) = L^{-\dagger}\delta f(Y)$ if $p > n$ (regression/overdetermined case), where $L^\dagger$ denotes the left inverse of A. Finally, a practical way to determine a set Y of previously evaluated points in the context of direct search is specified in the next section.

3.2. A numerical illustration

We have run Algorithm 1.1 (Direct Search) and Algorithm 1.2 (Gradient Method) on twenty problems from the CUTEr collection [4] of dimension $n = 20$. Algorithm 1.1 was applied with $\beta = 1/2$, $\gamma = 1$, and $D = D_\oplus$. The poll step was implemented in a cyclic way, where one starts using a direction in D stored one column after the column corresponding to the last direction used in the previous poll step. Algorithm 1.2 was applied with $c = 10^{-4}$ and $b = 1$. When running Direct Search, we applied Algorithms 3.3 and 3.4 to compute the indicator and trigger the moment of the switch to the Gradient Method, with $E = 20$ and $E_{\text{scaling}} = 5$. The threshold was set to $C_{\text{threshold}} = 2$. In the hybrid I^1/I^2 case, we have multiplied I_k^1 by 100 which has an effect equivalent to dividing I_k^2 by $D_2(n)^2 = 20$.

The sample set used to compute simplex gradients was selected as follows. First, we only computed simplex gradients after having at least $n + 1$ overall evaluations. If there is a need to compute an approximation to a gradient for indicator purposes when there are less than overall $n + 1$ evaluations, one applies finite differences; such a situation has rarely occurred but was anyhow taken into consideration when counting the number of function evaluations. At each iteration, the overall sample set of evaluated points is ordered by increasing distance to the current iterate. Then, one selects the sample set for the simplex gradient calculation as all the points within a distance of $5\alpha_k$ of the current iterate. If there are less than n points within this distance, we fill the set using the remaining ones until there are at least n. In such a way, we never compute underdetermined simplex gradients. If there are more than $n + 1$ points in the sample set, we solve the regression system in the least squares sense. In all cases (determined and overdetermined), the linear systems are solved by means of the QR factorization, and a regularization is applied to the diagonal elements of R, replacing them by $\text{sign}(R_{ii})10^{-8}$ when $|R_{ii}| < 10^{-8}$.

The results are shown first in Figs. 1 and 2 for the hybrid I^1/I^2 case which performed better than the pure I^1 one. In all cases, the budget of function evaluations for Direct Search was 500, except for problem `penalty1` where we went until 3000. The plots of the function values in Fig. 1 are given in a logarithmic scale ($\log(f - f_*)$ where f_* denotes the optimal value). Table 1 contains the relevant data corresponding to Fig. 1.

Although we report numbers of function and gradient evaluations, we remark that the primary goal of our work is not to develop a switched method more efficient or robust than both the derivative-free and the derivative-based underlying ones. Instead we tried to develop a technique that can identify an appropriated moment for the switch with desirable properties such as (i) not to switch when that can be foreseen as worse; (ii) switch when not enough progress is being achieved compared to the one that could be done if derivatives were available.

One can see that a switch occurred in 15 out of the 20 problems. Among the 5 problems where no switch occurred, in four of these (arglina, arwhead, nondia, and tquartic) direct search did better than gradient descent — a manifestation of a feature that is desirable for the indicators in question. The other one (`integreq`) is somehow the single failure of our approach in the sense that a switch should have occurred.

In an attempt to better balance the relative costs of a gradient calculation, it is reported in Table 2 the effort of the gradient and switched methods in the hybrid I^1/I^2 case for the problems for which the curve corresponding to switched one passed the one corresponding to gradient one (in the plots of Fig. 1). The effort is measured in two ways: (i) summing the number of function evaluations with the number of gradient evaluations multiplied by a factor of 5 (to match the case where the gradient is computed from automatic differentiation; see [6]); (ii) summing the number of function evaluations with the number of gradient evaluations multiplied by a factor of $n = 20$ (for the case where the gradient is approximated by finite differences). In 8 out of these 10 problems, the effort of the switched method was considerably less than that of the gradient method.

In Table 3, we provide more information about the relative effort of these two methods in the hybrid I^1/I^2 case for the remaining problems, meaning those not listed in Table 2 and for which there was a switch. Problem dqrtic is an example where depending on the way a gradient evaluation is weighted, it can be worth or not worth to use a switched version when compared to a derivative-based one. For the remaining problems (in particular for sparsine and vardim), it seems that the initial derivative-free phase was harmful. We note that numerical convergence would have been obtained for these two problems with a very large budget (see Fig. 4).

Similar figures (Figs. 3 and 4) and tables (Tables 4–6) are reported for the pure I^1 case. The major difference compared to the hybrid I^1/I^2 seems to be a lost of prediction of an appropriate switch. In fact, in the pure I^1 case, there was no switch in two other problems (freuroth and sinquad) where it would be appropriated to do it (see Tables 4–6).

4. Final remarks

One way to stop the run of a derivative-free method in the hope to switch to a faster, higher order method would be to look at the behavior in function values. In fact, looking at how the function decreases along the iterations would provide heuristic indicators to such a switch. However, stagnation in function values could occur for a number of reasons not necessarily because of the lack of derivatives. In this paper, we tried to propose a more systematic indicator for this switch grounded on certain theoretical principles related to known decrease properties. Our approach is certainly also of heuristic type as we compare lower bounds on the decreases attained by the derivative-free method to the ones that would be obtained if a derivative-based one would had been used. Note also that the indicators here depend on very few parameters, essentially on $C_{\text{threshold}}$, E, and $E_{\text{threshold}}$. Our experiments have shown us that the results presented here are relatively robust with respect to

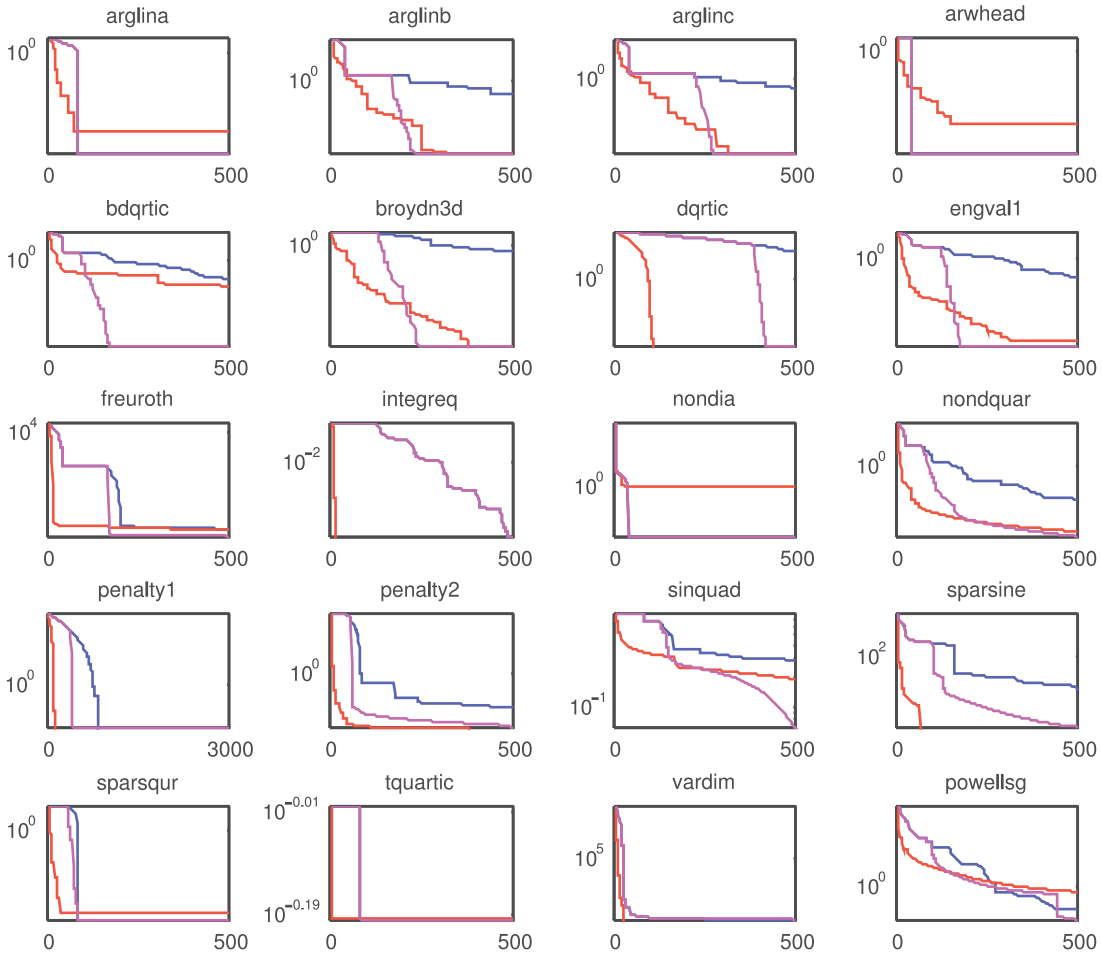

Fig. 1. Plots of functions values for Direct Search (solid, blue), Gradient Method (dash–dot, red), and Direct Search switched to Gradient Method in the hybrid I^1/I^2 case (dash, magenta). In the x-axis, we count function evaluations. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 1
Detailed results of Fig. 1 for Gradient Method (GM) and the Switched Method (SM) in the hybrid I^1/I^2 case. Columns 2–3: number of gradient evaluations. Columns 4–5: number of function evaluations. Columns 6–7: final value of f obtained.

	$\#\nabla f$ GM	$\#\nabla f$ SM	$\#f$ GM	$\#f$ SM	f GM	f SM
arglina	14	0	59	81	0.0000	0.0000
arglinb	25	15	456	226	4.6341	4.6341
arglinc	23	10	332	271	6.1351	6.1351
arwhead	14	0	157	39	0.0000	0.0000
bdqrtic	68	10	426	111	35.4101	35.4091
broydn3d	35	101	370	399	0.0000	0.0000
dqrtic	50	19	59	481	0.0000	0.0000
engval1	37	13	438	160	20.2787	20.2787
freuroth	87	3	409	168	2696.7893	2660.2755
integreq	8	0	66	500	0.0000	0.0001
nondia	29	0	459	37	1.9242	1.0000
nondquar	77	103	419	281	0.0092	0.0065
penalty1	96	852	1941	2148	0.0002	0.0002
penalty2	46	147	447	353	0.0090	0.0093
sinquad	69	72	424	273	0.1892	0.0660
sparsine	65	133	434	367	0.0002	1.1003
sparsqur	7	8	25	74	0.0000	0.0000
tquartic	7	0	66	83	0.6338	0.6328
vardim	23	103	474	397	0.0000	7.7142
powellsg	105	87	393	269	0.6369	0.0581

the changes in these parameters, especially in E and $E_{\text{threshold}}$. The choice of $C_{\text{threshold}}$ is understandably more critical and problem dependent but it also gives a user a tool for adjustment.

Our approach covers a common situation where the step size decreases in *successful* iterations in the usage of the derivative-free method, as long as (i) by *successful* here we mean that the trial point is accepted as the new iterate and (ii) a certain condition is explicitly or implicitly satisfied implying that the gradient at the current point is of the order of the step size. Then, such *successful* iterations will no longer be considered successful in the language

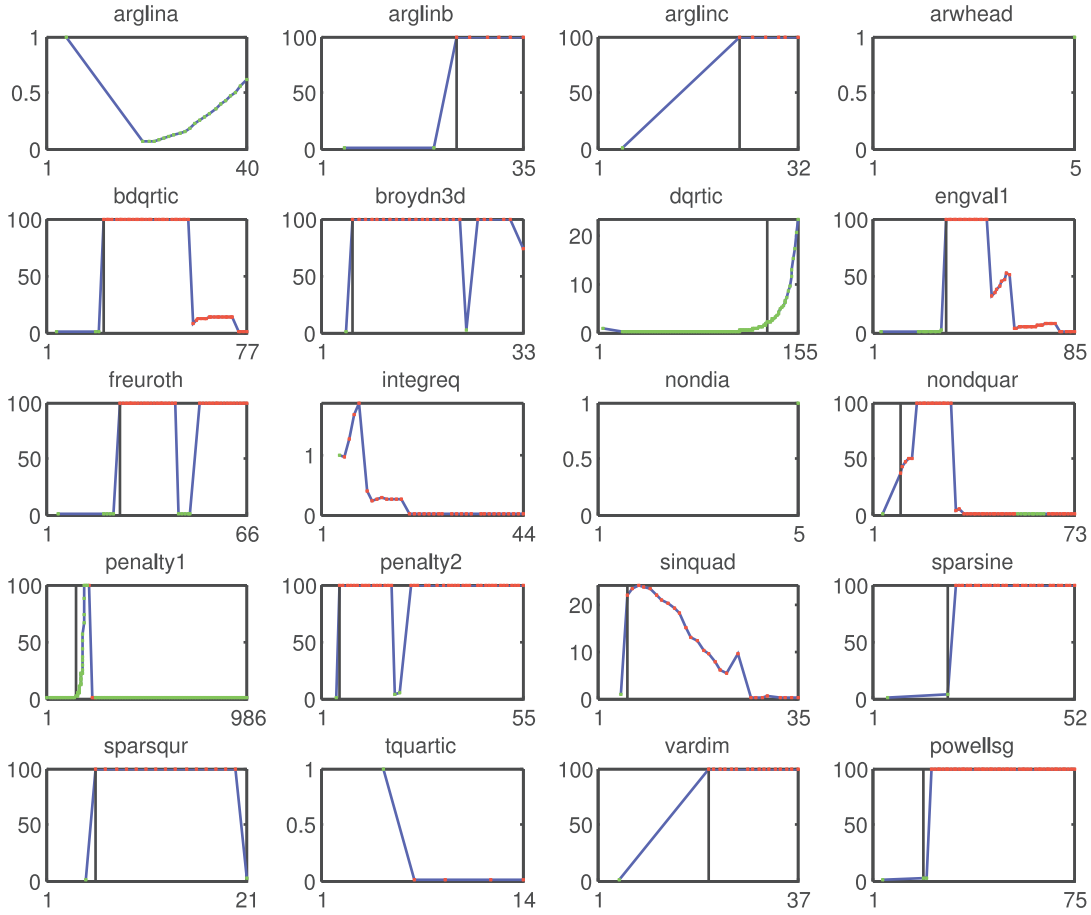

Fig. 2. Plots of the indicator values I_k/I_{scaling} in the hybrid I^1/I^2 case when they were calculated (points, in green or lighter for I_k^1 and in red or darker for I_k^2). The solid (blue) line is plotted for better visualization. The (black) vertical line signals the moment of the switch. Note that in the x-axis we count iterations, not function evaluations, and that all indicator values above 100 were plotted using this value for better visualization. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 2
It is reported the effort of the Gradient Method (GM) and the Switched Method (SM) in the hybrid I^1/I^2 case for the problems for which the curve corresponding to the SM passes the GM one in the plots of Fig. 1. Columns 2–3: sum of the number of function evaluations with the number of gradient evaluations multiplied by 5. Columns 4–5: sum of the number of function evaluations with the number of gradient evaluations multiplied by $n = 20$. Columns 6–7: value of f at the pass or right after it.

	+5 GM	+5 SM	+20 GM	+20 SM	f GM	f SM
arglinb	581	301	956	526	4.634146	4.634146
arglinc	447	321	792	471	6.135135	6.135135
bdqrtic	766	161	1786	311	35.410112	35.409468
broydn3d	545	904	1070	2419	0.000000	0.000000
engval1	623	225	1178	420	20.278662	20.278662
freuroth	844	183	2149	228	2696.789277	2679.938911
nondquar	804	796	1959	2341	0.009174	0.009108
sinquad	769	633	1804	1713	0.189190	0.187714
sparsqur	60	114	165	234	0.000000	0.000000
powellsg	918	704	2493	2009	0.636923	0.633652

Table 3
It is given more information about the Gradient Method (GM) and the Switched Method (SM) in the hybrid I^1/I^2 case for the remaining problems (those not listed in Table 2 and for which there was a switch). Columns 2–3: sum of the number of function evaluations with the number of gradient evaluations multiplied by 5. Columns 4–5: sum of the number of function evaluations with the number of gradient evaluations multiplied by $n = 20$. Columns 6–7: final value of f obtained.

	+5 GM	+5 SM	+20 GM	+20 SM	f GM	f SM
dqrtic	309	576	1059	861	0.000000	0.000000
penalty1	2421	6408	3861	19188	0.000163	0.000231
penalty2	677	1088	1367	3293	0.008987	0.009266
sparsine	759	1032	1734	3027	0.000227	1.100321
vardim	589	912	934	2457	0.000000	7.714170

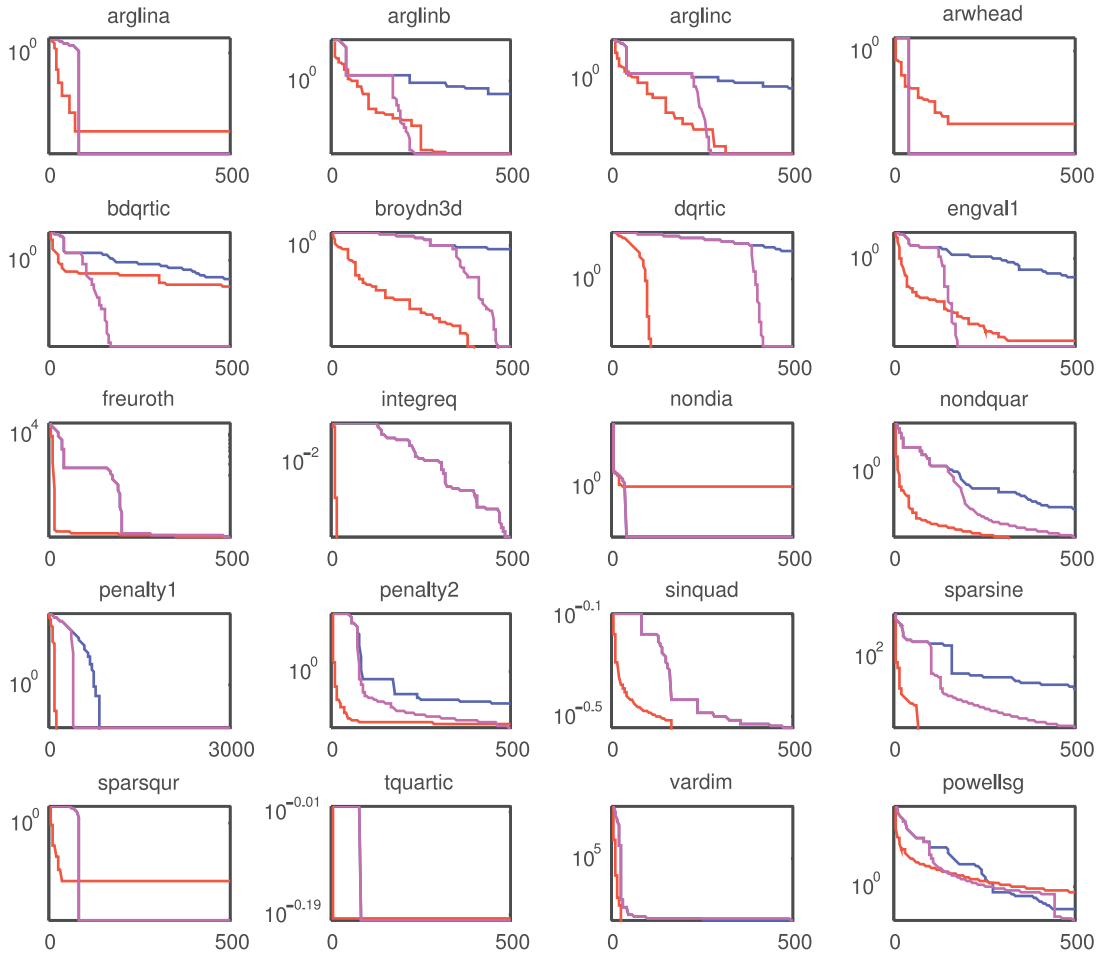

Fig. 3. Plots of functions values for Direct Search (solid, blue), Gradient Method (dash–dot, red), and Direct Search switched to Gradient Method in the pure l^1 case (dash, magenta). In the x-axis we count function evaluations. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 4
Detailed results of Fig. 3 for the Gradient Method (GM) and the Switched Method (SM) in the pure l^1 case. Columns 2–3: number of gradient evaluations. Columns 4–5: number of function evaluations. Columns 6–7: final value of f obtained.

	$\#\nabla f$ GM	$\#\nabla f$ SM	$\#f$ GM	$\#f$ SM	f GM	f SM
arglina	14	0	59	81	0.0000	0.0000
arglinb	25	15	456	226	4.6341	4.6341
arglinc	23	10	332	271	6.1351	6.1351
arwhead	14	0	157	39	0.0000	0.0000
bdqrtic	68	10	426	111	35.4101	35.4091
broydn3d	35	39	370	461	0.0000	0.0000
dqrtic	50	19	59	481	0.0000	0.0000
engval1	37	13	438	160	20.2787	20.2787
freuroth	87	0	409	500	2696.7893	2706.5165
integreq	8	0	66	500	0.0000	0.0001
nondia	29	0	459	37	1.9242	1.0000
nondquar	77	118	419	382	0.0092	0.0152
penalty1	96	852	1941	2148	0.0002	0.0002
penalty2	46	129	447	335	0.0090	0.0088
sinquad	69	0	424	500	0.1892	0.2890
sparsine	65	133	434	367	0.0002	1.1003
sparsqur	7	1	25	81	0.0000	0.0000
tquartic	7	0	66	83	0.6338	0.6328
vardim	23	103	474	397	0.0000	7.7142
powellsg	105	87	393	269	0.6369	0.0581

of our proposed approach and will be part of those where the step size is reduced.

In particular, the indicators proposed in this paper for the switch from a derivative-free method to a derivative-based one can also be used if the former is a trust-region method. One would not count in C_k iterations where the point and the trust-region radius do not change, like in a model-improvement iteration or in a critical iteration of that type. On the other hand, r_k may correspond to an unsuccessful iteration, an acceptable iteration

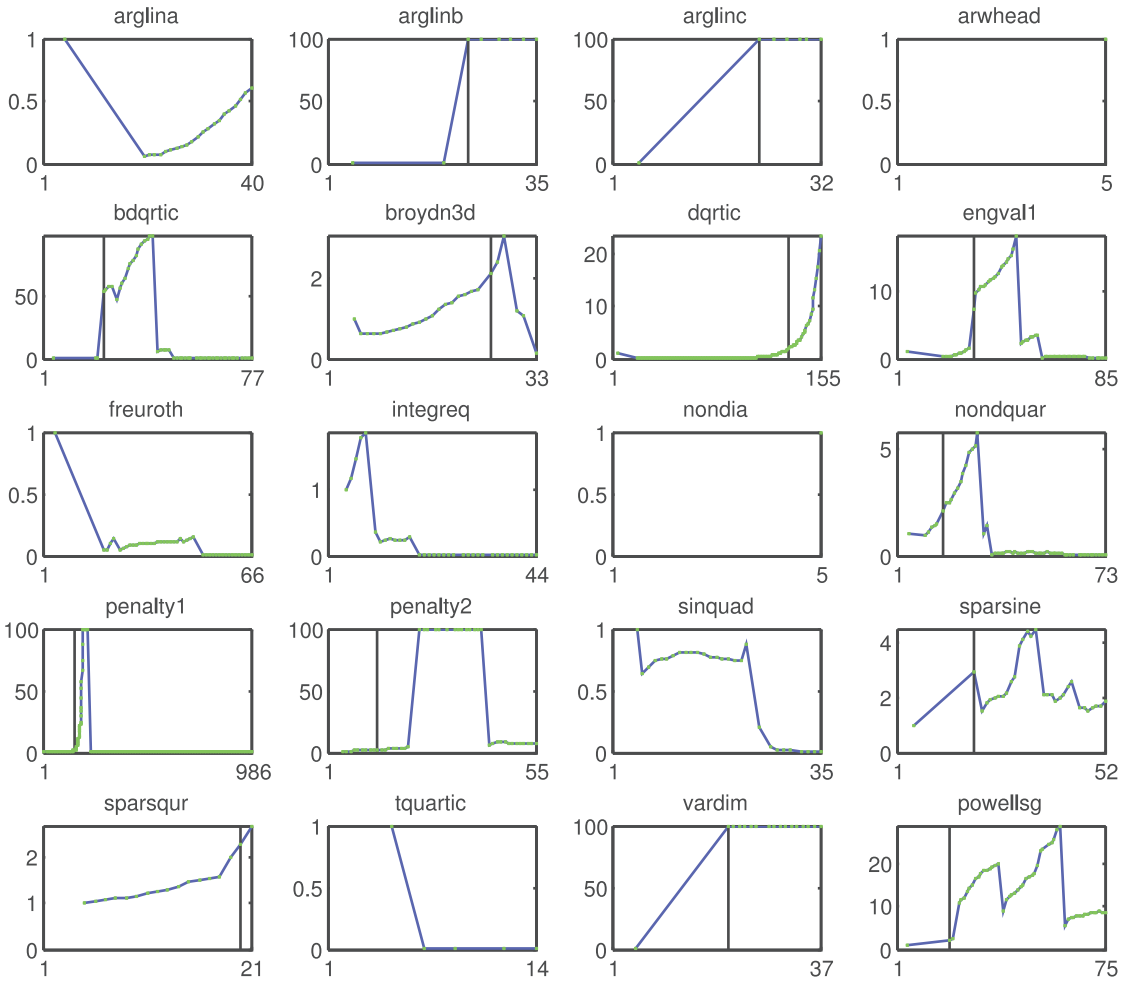

Fig. 4. Plots of the indicator values I_k/I_{scaling} in the pure I^1 case when they were calculated (points, in green). The solid (blue) line is plotted for better visualization. The (black) vertical line signals the moment of the switch. Note that in the x-axis we count iterations, not function evaluations, and that all indicator values above 100 were plotted using this value for better visualization. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 5
It is reported the effort of the Gradient Method (GM) and the Switched Method (SM) in the pure I^1 case for the problems for which the curve corresponding to the SM passes the GM one in the plots of Fig. 3. Columns 2–3: sum of the number of function evaluations with the number of gradient evaluations multiplied by 5. Columns 4–5: sum of the number of function evaluations with the number of gradient evaluations multiplied by $n = 20$. Columns 6–7: value of f at the pass or right after it.

	+5 GM	+5 SM	+20 GM	+20 SM	f GM	f SM
arglinb	581	301	956	526	4.634146	4.634146
arglinc	447	321	792	471	6.135135	6.135135
bdqrtic	766	161	1786	311	35.410112	35.409468
broydn3d	545	656	1070	1241	0.000000	0.000000
engval1	623	225	1178	420	20.278662	20.278662
freuroth	844	500	2149	500	2696.789277	2706.516502
nondquar	804	972	1959	2742	0.009174	0.015228
sinquad	769	500	1804	500	0.189190	0.288975
sparsqur	60	86	165	101	0.000000	0.000000
powellsg	918	704	2493	2009	0.636923	0.633652

(where the ratio between actual and predicted reductions is positive but too small), or a critical iteration of the other type, where in all cases the trust-region radius is reduced (see the details in [3]).

As for the derivative-based method, one can use any gradient-based algorithm, of first or second order type, for which the decrease in functions values can be estimated as a function of the size of the gradient. The derivation of complexity bounds for derivative-based trust-region methods (as done in [5]) is also based on a condition like (5).

Acknowledgment

Support for the third author was partially provided by FCT under grants UID/MAT/00324/2013, SFRH/BSAB/114622/2016, and P2020 SAICTPAC/0011/2015.

Table 6
It is given more information about the Gradient Method (GM) and the Switched Method (SM) in the pure I^1 case for the remaining problems (those not listed in Table 5 and for which there was a switch). Columns 2–3: sum of the number of function evaluations with the number of gradient evaluations multiplied by 5. Columns 4–5: sum of the number of function evaluations with the number of gradient evaluations multiplied by $n = 20$. Columns 6–7: final value of f obtained.

	+5 GM	+5 SM	+20 GM	+20 SM	f GM	f SM
dqrtic	309	576	1059	861	0.000000	0.000000
penalty1	2421	6408	3861	19 188	0.000163	0.000231
penalty2	677	980	1367	2 915	0.008987	0.008928
sparsine	759	1032	1734	3 027	0.000227	1.100321
vardim	589	912	934	2 457	0.000000	7.714170

References

[1] A.R. Conn, K. Scheinberg, L.N. Vicente, Introduction to Derivative-Free Optimization, in: MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2009.

[2] M. Dodangeh, L.N. Vicente, Worst case complexity of direct search under convexity, Math. Program. 155 (2016) 307–332.

[3] R. Garmanjani, D. Júdice, L.N. Vicente, Trust-region methods without using derivatives: Worst case complexity and the non-smooth case, SIAM J. Optim. 26 (2016) 1987–2011.

[4] N.I.M. Gould, D. Orban, P.L. Toint, CUTEr, a constrained and unconstrained testing environment, revisited, ACM Trans. Math. Software 29 (2003) 373–394.

[5] S. Gratton, A. Sartenaer, Ph.L. Toint, Recursive trust-region methods for multiscale nonlinear optimization, SIAM J. Optim. 19 (2008) 414–444.

[6] Andreas Griewank, On automatic differentiation, in: Mathematical Programming, Applications, Kluwer Academic Publishers, Amsterdam, 1989, pp. 83–108.

[7] T.G. Kolda, R.M. Lewis, V. Torczon, Optimization by direct search: New perspectives on some classical and modern methods, SIAM Rev. 45 (2003) 385–482.

[8] Y. Nesterov, Introductory Lectures on Convex Optimization, Kluwer Academic Publishers, Dordrecht, 2004.

[9] J. Nocedal, S.J. Wright, Numerical Optimization, second ed., Springer-Verlag, Berlin, 2006.

[10] L.N. Vicente, Worst case complexity of direct search, EURO J. Comput. Optim. 1 (2013) 143–153.

[11] J. Virieux, S. Operto, An overview of full-waveform inversion in exploration Geophysics, Geophysics 74 (2009) WCC1–WCC26.